Rethinking Conventional Collaborative Filtering for Recommending Daily Fashion Outfits

Anders Kolstad, Özlem Özgöbek, Jon Atle Gulla
Norwegian University of Science and Technology
Trondheim, Norway
andekol@stud.ntnu.no
{ozlem.ozgobek,jon.atle.gulla}@ntnu.no

Simon Litlehamar
Accenture AS
Fornebu, Norway
simon.litlehamar@accenture.com

ABSTRACT

A conventional collaborative filtering approach using a standard utility matrix fails to capture the aspect of matching clothing items when recommending daily fashion outfits. Moreover, it is challenged by the new user cold-start problem. In this paper, we describe a novel approach for guiding users in selecting daily fashion outfits, by providing outfit recommendations from a system consisting of an Internet of Things wardrobe enabled with RFID technology and a corresponding mobile application. We show where a conventional collaborative filtering approach comes short when recommending fashion outfits, and how our novel approach—powered by machine learning algorithms—shows promising results in the domain of fashion recommendation. Evaluation of our novel approach using a real-world dataset demonstrates the system's effectiveness and its ability to provide daily outfit recommendations that are relevant to the users. A non-comparable evaluation of the conventional approach is also given.

CCS CONCEPTS

•**Information systems** →**Evaluation of retrieval results;** *Web applications;* •**Computing methodologies** →*Classification and regression trees;*

KEYWORDS

Recommender Systems, Machine Learning, Fashion Recommendation, Collaborative Filtering, Internet of Things

1 INTRODUCTION

Selecting an outfit every morning is a task that many people struggle with, often due to time constraints or the feeling of having nothing to wear. In [20], Pruit argues that our selection of an outfit influences other people's impressions of us, and that it is of high importance to our cultural lives. Moreover, the average Norwegian has 359 unique garments in their closets [15]. This suggests that people need guidance and suggestions for selecting an outfit from their clothing haystack each morning.

RecSysKTL Workshop @ ACM RecSys '17, August 27, 2017, Como, Italy

Klepp and Laitala found that 20% of clothes bought by Norwegians were never or rarely used [15]. A reason for this might be that they did not actually like the item they bought or that the item did not match any existing clothing items in their closet. This information is very valuable to the clothing retailer. With such information, the retailer can map the customer's taste profile and generate targeted ads for the customer, reducing the number of unnecessary purchases, and increasing the number of satisfied customers.

Generating such outfit suggestions and targeted ads can be made a reality by recommender systems. A recommender system tries to predict the rating value of a user-item combination, where the user has indicated their ratings for other items in the past [1]. The system tracks these ratings by receiving user feedback. User feedback is classified into *explicit* and *implicit*. Explicit feedback is when the user explicitly rates an item on, e.g., a 5-star scale. Implicit feedback records other user interactions, e.g., how long a user spends on a web page on a certain topic. With the retrieved ratings by user feedback, the recommender system can predict the user's ratings of new items, and suggest the items with a high predicted rating. One of the most successful recommendation technique is called collaborative filtering (CF) [22]. CF recommends items on the assumption that users who have interacted in similar ways before, will have common interests in the future as well. Conventional CF bases its recommendations from a matrix called the *utility matrix*, which captures every rating value for the user-item combinations known to the system [17]. Table 1 shows an example of such a matrix, consisting of user-item combinations of users and movies. A known challenge in CF is called new user cold-start problem. This challenge is about how to recommend items to new users that have not rated any items yet. Suppose we were to introduce a fourth user in Table 1. The user-item combinations for this fourth user would all have '?' as a value. How to then recommend items to this user is not an easy task.

Table 1: Example of a utility matrix.

	Titanic	The Godfather	Pulp Fiction	The Notebook
Alice	5	2	?	?
Bob	2	?	4	1
Charlie	4	1	5	4

Recommending individual items, such as in Table 1, is what nearly all recommender systems are focusing on. In recent years, recommendations of collections, such as music playlists [12, 13, 23],

has gained a lot of attention. Hansen and Golbeck identified some key aspects that affects the recommendation of collections [10]. One aspect that especially applies to outfit recommendation is the co-occurrence interaction effect. Matching clothing items (items that go well together) will have a positive interaction effect when they co-occur together, and will therefore generate a more relevant outfit recommendation to the user.

In [16], we proposed Connected Closet, a system consisting of an Internet of Things wardrobe enabled with an RFID reader, so that clothing items with RFID tags can be checked in and out of the closet, generating implicit feedback on clothing items the user likes. Using a mobile application, the user can give explicit feedback on outfit he likes, and receive daily outfit recommendations based on outside temperature and wardrobe inventory. In this paper, we describe an implementation of the proposed system. We show where a conventional CF approach comes short in terms of the new user cold-start problem and where it fails to capture the co-occurrence effect between items. Moreover, we propose a novel CF approach that mitigates the shortcomings of the conventional approach and implement the novel approach into the proposed system. Evaluations using a real-world dataset are performed on both approaches.

The main contributions of this paper are:

(1) A novel CF approach for recommending daily fashion outfits.
(2) An accuracy evaluation of the approach using different classification algorithms.

This work is a joint effort between the Smartmedia program[1] at NTNU[2] and Accenture Norway[3]. The Smartmedia program is researching mobile context-aware recommender systems. While, in this work, Accenture's main goal is to research modern technology for building web-based information systems and to keep track of technology key trends, such as Internet of Things.

The rest of the paper is structured as follows. In Section 2, we give an overview of related work, followed by a description of the proposed system in Section 3. Section 4 introduces the concept of outfit recommendation. The recommendation approaches are described in Section 5 and Section 6. Evaluation of the approaches is given in Section 7. We conclude with a summary and discuss future work in Section 8.

2 RELATED WORK

There are not many systems addressing daily outfit recommendations from either an Internet of Things wardrobe or a virtual wardrobe. In this section, we give an overview of the state of the art, identify gaps in these works, and show where our system differs from past work and how it complements previous work.

Dumeljic et al. propose a virtual wardrobe implemented as a mobile application [6]. By explicitly stating the user's current mood, the user can add clothing items that best fit the mood, to the virtual inventory. In [6], the outfit recommendation approach is not described and has not been implemented in the system. Moreover, a user study of ten people was conducted, where they concluded that mood is a motivator for selecting outfits, but that users would be more invested in the system if it also considered weather.

In [19], Limaksornkul et al. also propose a mobile application used as a virtual wardrobe. They try to solve the problem of efficiently managing closet inventory and guiding users in selecting clothes based on the user's fashion style, trends, their friends' styles, weather, and occasion. In the mobile application, the users can manage their clothes, and receive statistical-based, weather-based, and event-based clothing suggestions. The statistical-based recommendation engine is preliminary and is the only approach that takes user's preferences into account. Moreover, no evaluation of the system is given.

A smart wardrobe system is proposed by Goh et al. in [9]. Here, garments attached to RFID tags can be scanned in the user's closet. Using a system application, the user can get clothing recommendations based on the user's mood, preferred color or and occasion.

Yu-Chu et al. propose a recommendation system using a modified Bayesian network for generating outfit recommendations from the user's clothing items enabled with RFID tags stored in a smart wardrobe [24]. By taking weather, season, and occasion into consideration, the system first select a top, and then finds bottoms which match the selected top. The process of selecting a bottom depend on user feedback rating the combination. An experiment on 10 users concluded that the proposed system gave more satisfied users than a baseline using a basic Bayesian network without user feedback.

An important aspect that needs to be mentioned is that virtual wardrobes are heavily dependent on explicit user feedback, while the Internet of Things wardrobes can make use of implicit user feedback as well.

As seen in the works above, most of the recommender systems are preliminary, and does not contain clear steps for the recommendation algorithm. The ones that do have an implemented recommender system only have user studies and are lacking accuracy evaluation of their recommendations. In this paper, we describe a fully implemented prototype, using similar architecture to [9] and [24], enabled with a novel recommendation approach evaluated on a real-world dataset. To the best of our knowledge, our novel approach is a completely unique way of generating recommendations using CF. This is mostly because the majority of CF recommender systems today, are heavily based on the utility matrix [22], which is not present in our approach.

3 SYSTEM OVERVIEW

In this section, we describe the architecture of the smart wardrobe proposed in [16]. Moreover, we explain how the users receive recommendations through the mobile application which is a part of the architecture. We built and implemented a prototype of the whole system and created a short demonstration video available at https://goo.gl/rZBZqo.

3.1 Architecture

Figure 1 shows a high-level view of the architecture. The Closet is embedded with a Raspberry PI[4] connected to an RFID reader. Clothing items enabled with an RFID tag and that has their id number stored in the Cloud, are clothing items that are compatible

[1] http://research.idi.ntnu.no/SmartMedia/
[2] http://www.ntnu.edu/
[3] https://www.accenture.com/no-en

[4] A tiny computer. See https://www.raspberrypi.org/

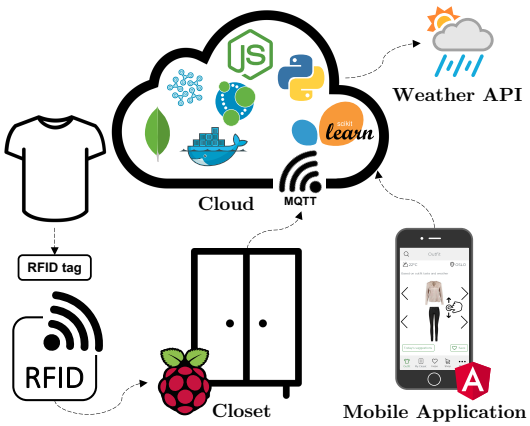

Figure 1: High level architecture.

with the system. Such clothing items can be manually scanned through the RFID reader. When a scanning occurs, a message gets broadcasted to multiple services deployed in the Cloud. These services include—among others—a recommender service and an inventory service. By communicating with each other and a third-party Weather API, they provide outfit recommendations to the Mobile Application.

3.2 Mobile Application

When the user opens the mobile application, he gets displayed a recommendation for an outfit that suits today's temperature and is inside the user's closet. By swiping through a list, the user is displayed multiple recommended outfits. Moreover, the user can modify the recommended outfit by using the arrows that corresponds to each clothing item. By clicking a Save button, the user gives an explicit positive feedback on the displayed outfit, indicating that the user has this outfit as one of his favorites.

Figure 2: Screenshot of the mobile application.

4 OUTFIT RECOMMENDATION

We define an outfit, denoted o, as a tuple of two items, c_1 and c_2, where c_1 is a top and c_2 is a bottom. Although clothing outfits can also contain more, or less, than two items, the current version of our system only addresses outfits of two items. This is with the assumption that most outfits comprise of one top and one bottom. Recommendation of outfits consisting of a one-piece, e.g., a dress, or with additional accessories, is planned for later research.

4.1 Inclusion Criteria

To ensure that the user receives outfit recommendations that are relevant for a given day, we define an inclusion criteria for the clothing items that can be part of a recommended outfit. The inclusion criteria are defined as follows:

(1) **Clothing item must be inside the closet.** The status of the item is determined by the latest RFID tag scan.
(2) **Clothing item must be suitable for current weather.** Items are stored in a database with a suitable temperature range property. This is the range of temperatures a clothing item is comfortable to wear. The outside temperature at time of recommendation, must be inside the item's suitable temperature range.

All clothing items that are owned by a user u_i and fits the inclusion criteria is represented as a set $I(u_i)$. All outfit combinations that can be generated from $I(u_i)$ are added to the set $O(u_i)$.

4.2 User Ratings

The favored outfits indicated (explicitly or implicitly) by the user, are stored in the system using unary positive-only values. Outfits that have not been rated are outfits that the users either do not like or have not been seen or used together from the user's closet $C(u_i)$. Not rated outfits will be referred to as 'neutral' outfits in the rest of this paper.

4.3 Recommended Outfits

The list of recommended outfits that the user receives in the mobile application is generated by the system's recommender service that returns the set $R(u_i)$ of recommended outfits for the user.

4.4 Notation

All the notations defined in this section are summarized in Table 2. These notations will be used throughout the paper.

Table 2: Notations used in this paper.

Notation	Description
u_i	The ith user (owner) of a closet.
c_j	The jth clothing item.
$o_k = (c_1, c_2)$	An outfit of c_1 and c_2.
$C(u_i) = \{c_1, \ldots, c_l\}$	Every clothing items the user owns.
$I(u_i) = \{c_1, \ldots, c_m\}$	Clothes fitting the inclusion criteria.
$O(u_i) = \{o_1, \ldots, o_n\}$	Outfit combinations of items in $I(u_i)$.
$R(u_i) = \{o_1, \ldots, o_p\}$	Outfits recommended to the user.

5 RETHINKING CONVENTIONAL CF

In this section, we introduce an approach for outfit recommendation using a conventional utility matrix for collaborative filtering. We discuss where this approach comes short, and introduce a novel approach for outfit recommendation using an outfit-item matrix.

5.1 Conventional CF Approach

An obvious solution to recommending fashion outfits is to map the users' favorite outfits onto a utility matrix U, consisting of users and outfits. Then, using a neighborhood model, one could predict new outfits for users by comparing the user's interaction pattern with users with same interaction pattern. To recommend the daily outfits $R(u_i)$, we need to match the predicted outfits with the items that fit the inclusion criteria $I(u_i)$, and filter out outfits that do not contain only such items. The approach is illustrated in Figure 3.

The first problem with this approach is that it can only recommend outfits that have been favored by other users. In other words, it cannot generate completely new outfits, and therefore fails to capture the co-occurrence effect between individual items. Another problem with this approach is that it is challenged by the new user cold-start problem. Users who have not favored any outfits or checked out any items, cannot receive recommendations. Lastly, privacy is becoming a huge concern in recommender systems [2, 3], and in this approach, we store all the users' ratings in one centralized matrix, causing a huge risk for the users' privacy.

5.2 Novel Outfit-Item Matrix Apprach

By basing our recommendations on the idea that users that have similar items in their closets will also have similar taste in outfits, we propose a novel approach where we rethink the conventional approach by completely transforming the utility matrix. In Figure 4, we create a matrix Z, where the columns represent outfits, and the rows represent the clothing items that compose the outfit. Each outfit is associated with a weight w. This weight is the number of users who have favored an outfit. Using Z and W, we train a classifier using a classification model. Outfits that have been favored by users and have an associated weight above 0 will be classified as 'positive', while outfits with an associated weight of 0 will be classified as 'neutral'. When the model has been trained, we generate all the possible outfit combinations $O(u_i)$, of the items that fit the inclusion criteria for the given user u_i. By using the classifier, we can now recommend the outfits that are classified as 'positive' to the user $R(u_i)$.

The advantages of this approach are that it captures the co-occurrence interaction effect between two clothing items. This is because it considers the clothing items that an outfit is composed of, instead of just looking at the outfits as a whole. Moreover, it is not challenged by the new user cold-start problem because we assume that people that own similar clothing items will have same taste in outfits as well. Lastly, this approach has a huge advantage in terms of user privacy, because it does not need to store the user-item combinations in one centralized matrix.

In Figure 5, we give an example of a possible recommendation pipeline that can occur in our system using the novel approach. To the left is the set of all the clothing items owned by the user. By inputting this and the current outside temperature at the user's location, the function f_1 filters out and generates possible outfits for recommendation wrt. the inclusion criteria. These outfits are then inputted to f_2, which follows the same steps as described in Figure 4. In the end of the pipeline, we get the generated set of recommended outfits that is displayed in the mobile application.

Although not implemented in our system, this approach could be easily used by a clothing retailer to generate targeted ads by inputting clothing items from the retailer together with the user's clothing items in $C(u_i)$. Then, the clothing retailer could recommend new outfits that the users might want to buy, or individual items that would make a great outfit with clothing items already owned by the user.

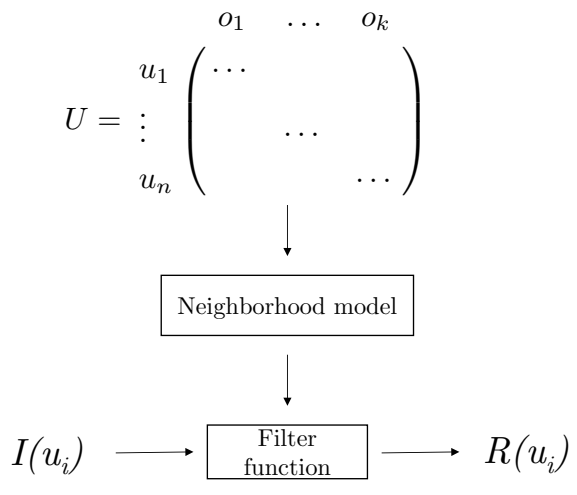

Figure 3: Conventional CF approach using a utility matrix.

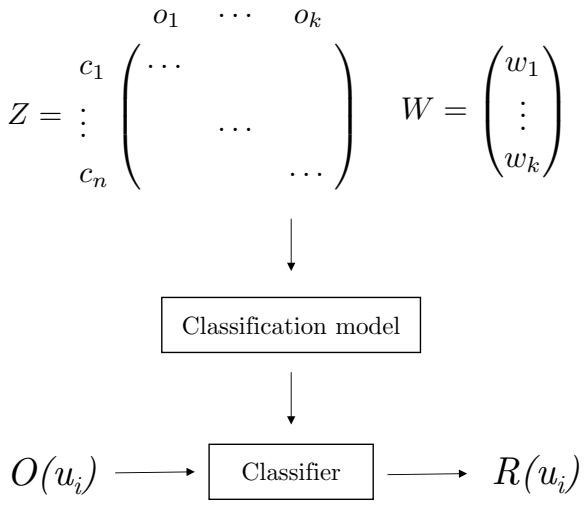

Figure 4: Novel approach using an outfit-item matrix.

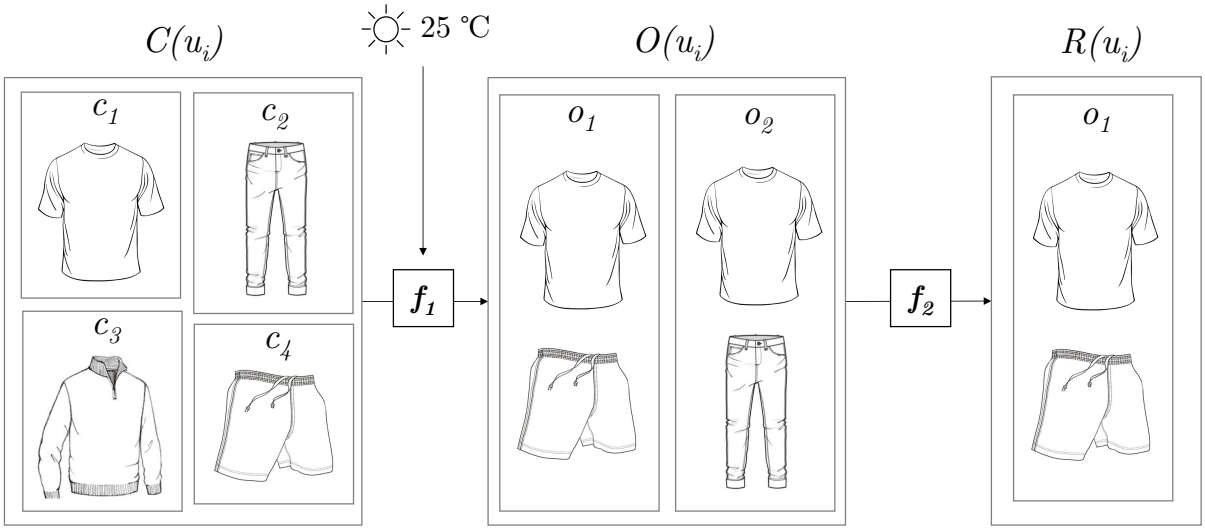

Figure 5: Example of a possible recommendation pipeline using the novel approach.

6 RECOMMENDATION MODEL

In this section, we present the recommendation model for our novel approach using different classification models. The chosen classification models are widely known and perform well in many domains [4, 5]. The classification models also include a baseline classifier. Moreover, we introduce some neighborhood models that are applied with the conventional approach.

6.1 Classification Models

Naïve Bayes. Assuming the attributes of the samples are conditionally independent and given the sample's class labels, Naïve Bayes assigns a test sample the class label Y by maximizing the numerator in this equation [18]:

$$P(Y \mid \boldsymbol{X}) = \frac{P(Y)\prod_{i=1}^{d} P(X_i \mid Y)}{P(\boldsymbol{X})}, \tag{1}$$

where $\boldsymbol{X}$ is a set of d attributes.

Adaptive Boosting (AdaBoost). Over the recent years, classification techniques known as ensemble methods have gained a lot of attention. One of the most popular ones is AdaBoost. It aggregates over a set of weak learners $h_t(x)$ that tends to perform slightly better than a random classifier. The final classifier $H(x)$ is then obtained by ensembling the weak learners by a weighted majority voting scheme using this equation [7]:

$$H(x) = sign\Big(\sum_{t=1}^{T} \alpha_t h_t(x)\Big), \tag{2}$$

where α_t is the assigned weight for each weak learner.

To pick the weak learners, each training sample is associated with a weight indicting its importance. AdaBoost will then pick its weak learners in a forward stage-wise manner by focusing on predicting the high-weight samples correctly.

Gradient Boosting. Another popular ensemble method that relies on a set of weak learners is called Gradient Boosting. It follows the same fundamental idea as AdaBoost, but instead of focusing on the sample weights when picking its weak learners, it focuses on gradients [8].

Uniform. As a baseline, we use a classifier that generates class predictions uniformly at random.

6.2 Neighborhood Models

To predict the ratings of the user-outfit combinations in the matrix U, given in Figure 3, we apply the user-based neighborhood model [1]. This model predicts user ratings by finding users that have rated similar outfits. To find similar users, we can apply different similarity measures. In our model, we apply Jaccard (JAC) and cosine similarity (COS) as defined by Equation 3:

$$Sim_{JAC}(\boldsymbol{A}, \boldsymbol{B}) = \frac{|\boldsymbol{A} \cap \boldsymbol{B}|}{|\boldsymbol{A} \cup \boldsymbol{B}|} \qquad Sim_{COS}(\boldsymbol{A}, \boldsymbol{B}) = \frac{\boldsymbol{A} \cdot \boldsymbol{B}}{||\boldsymbol{A}||\,||\boldsymbol{B}||} \tag{3}$$

After user similarities have been calculated we can predict the ratings $\hat{r}_{ui}$ of unrated outfits using this formula:

$$\hat{r}_{ui} = \frac{\sum_v Sim(u,v) r_{vi}}{\sum_v |Sim(u,v)|} \tag{4}$$

6.3 Ranking Model

To rank the outfits that are predicted to the user in $R(u_i)$, using the novel approach, we assign each prediction of an outfit o_j to a ranking score equal to the classifier's probability of the class label being 'positive' $P(w_j > 0 \mid o_j)$. It should be noted that this is not a personalized ranking model, but as seen from our results, it performed well for each individual user.

The conventional approach does not use classification models, so the probability of the predicted class label is not available. Instead, the outfits are ranked according to the predicted rating calculated using the similarity measures.

7 EXPERIMENTS

In this section, we describe the setting for how our experiment was performed. We give a detailed description of the dataset that was used and present the results of the different models that were evaluated. The main goals of the experiments are to demonstrate the effectiveness of the system and to compare and select the best classification model for our system.

7.1 Dataset

The dataset is scraped from Polyvore.com[5]. Polyvore is a social media site where users can create clothing outfits by matching individual clothing items. Other users can then 'like' these outfits by a clicking a 'like button'.

From the available outfits at Polyvore, we first gathered the most liked outfits from the last 3 months. For these outfits, we filtered the outfits so that they only contained a top and a bottom. Then, we collected other outfits that these items also were a part of, and filtered them. Lastly, we gathered all the user likes for each of the outfits we had gathered. Table 3 describes the size of the dataset.

Table 3: Data statistics on Polyvore dataset.

# Outfits	# Clothes	# Users	# Likes
6,186	158	7,093	19,287
Positive: 260	**Tops:** 81		
Neutral: 5,917	**Bottoms:** 87		

From the gathered dataset, we have 260 outfits that are classified as 'positive' and 5,917 that are classified as 'neutral'. This means that the dataset has an imbalance approximately of 23 to 1.

In total, there are 158 individual clothing items in the dataset. This means that the feature vectors used in the classification models will be relatively sparse binary vectors of 158 dimensions.

7.2 Evaluation Methods

To evaluate our novel approach, we iterated through the following procedure for all users with at least 20 outfit likes: For all of the user's favorite outfits, we hide each of the user's ground-truth favorite outfits from the system by decreasing the outfits' corresponding weights in W by 1. Then, we train the classification model using Z and W. Moreover, with the assumption that a user only own items that are part of the items the user likes, we generate outfit combinations, assuming all of the items in $C(u_i)$ fitting the inclusion criteria. We then compared the predicted class labels of the generated outfits combinations to the true favorite outfits of the user. We also ran the procedure a second time, but now by randomly removing 50% of the users' tops and bottoms in $C(u_i)$. This was done to simulate outfit recommendations from a half empty closet. In Table 4, we summarize some statistics for the test sets that was generated by running these methods. As seen in this table, there are—on average—quite many outfits that are being classified for each user $O(u_i)$, compared to the true number of the user's favorite outfits $O(u_i)^{TP}$.

[5]http://www.polyvore.com/

Table 4: The average properties for the users in the test sets.

Closet size	$avg(\|O(u_i)\|)$	$avg(\|O(u_i)^{TP}\|)$
Full	682.5	31.4
Half empty	164.0	17.6

To reduce the dimensionality of the samples and to detect items that are interrelated, the multivariate analysis technique called *principal component analysis* was applied to the samples before training the models [14]. The reduction is done by transforming to a new set of uncorrelated features ordered so that the first ones retain most of the original variation.

For evaluating the conventional approach using the different neighborhood models, we first randomly removed 30% of the user likes from the utility-matrix. Then, we predicted all outfit likes for each user, and filtered them out wrt. $I(u_i)$ using the same assumption above. The recommended outfits were then compared to the true outfit likes.

7.3 Evaluation Metrics

If we look at the task of recommending the outfits as retrieving all relevant items (outfits) from a collection of outfits separated into the two classes; relevant and not relevant, we can apply the popular accuracy metrics from information retrieval systems. In our case, we say that the relevant outfits are the ones classified as 'positive', and the not relevant are the outfits classified as 'neutral'. Then, we can use a popular metric known as Recall. It measures the ratio of relevant items retrieved to the number of all relevant items available [11]:

$$\text{Recall} = \frac{|\text{relevant items retrieved}|}{|\text{all relevant items}|} \tag{5}$$

In this paper, we also report Recall@N, which is the Recall in a ranked list just considering the N first elements. We compute Recall and Recall@N by averaging over the result for each user u_i.

A way to graphically display the tradeoff between the true positive rate and the false positive rate, is known as a receiver operating characteristic (ROC) curve. The true positive rate is the same as Recall, and the false positive rate is the ratio of non-relevant items retrieved to the number of all non-relevant items available. The ROC curve is great to compare the performance difference between classifiers, where the best classifiers tend to be located in the upper left corner of the diagram. The classifiers that performs best on average will have a large area under the ROC curve (AUC) [11].

To evaluate the ranking via utility, we sum the utility of an outfit j to a user u over a ranked recommended list of size L. By summing over this value for each user, we obtain the R-score as follows [1]:

$$\text{R-score} = \sum_{u=1}^{m} \sum_{j \in I_u,\, v_j \leq L} \frac{max\{r_{uj}, 0\}}{2^{(v_j - 1)/\alpha}}, \tag{6}$$

where v_j is the rank of outfit j and r_{uj} is the ground-truth rating of outfit j. α is the half-life, set to 5 in our experiments. The higher the R-score is, the true favorite outfits for each user tend to appear in the top of the ranked list.

Table 5: Results from evaluation of novel approach.

Model	Overall AUC	Overall Accuracy	Overall Recall	Top-L R-score	Top-L Recall@5	Top-L Recall@10	Top-L Recall@15	Top-L Recall@20
Naïve Bayes	.704	.870	.756	566.2	.091	.183	.287	.382
Gradient Boosting	.864	.878	.997	851.9	.111	.228	.337	.448
AdaBoost	.885	.723	.978	872.1	.113	.223	.334	.442
Uniform	.500	.500	.493	88.0	.024	.052	.077	.095

7.4 Results and Discussion

In this section, we present our results and discuss some insight we obtained while running the experiments. By the end of this section we will have answered the following questions:

- Q1. How do the different classification models compare using our novel approach?
- Q2. How does closet size affect the recommendation results?
- Q3. To what extent can the conventional approach be used to recommend new outfits to the users?

The evaluation method for the novel approach was performed using the classification models in Section 6.1. For Naïve Bayes the best configuration was setting a prior probability for the 'neutral' class label to 0.99 and a 0.01 prior probability for the 'positive' class. This was mostly due to the 23 to 1 imbalance in the dataset. AdaBoost gave the best result using decision trees as weak learners and with a learning rate of 1.0. Gradient Boosting performed best with similar configurations.

In Table 5, we report AUC, Accuracy and Recall for the predicted class labels for all of the outfits that were tested when simulating a full closet. In the right-hand side of the table, we also report the R-score and Recall@N in a ranked list of L outfits. Because each user has different numbers of clothes in their closet, every user is recommended a ranked list of various lengths of L. The best performing model in each category is highlighted by underlining its result. As seen in the table, Gradient Boosting and AdaBoost are the dominating models in all categories. On average and overall, Gradient Boosting performs best, while in a top-L ranked list, AdaBoost performs slightly better. For N > 5, Gradient Boosting was—at maximum—only .006 points better than AdaBoost in Recall@N. In terms of the R-score, AdaBoost is superior to Gradient Boosting. Because of this, we conclude that AdaBoost is the model yielding highest utility to the users.

In Figure 6, we plot a ROC curve for the different models used to generate a single ranked list of user-outfit pairs. This type of ROC curve is sometimes referred to as a global ROC curve [21]. As indicated by the gray dotted line, AdaBoost is the best model at a

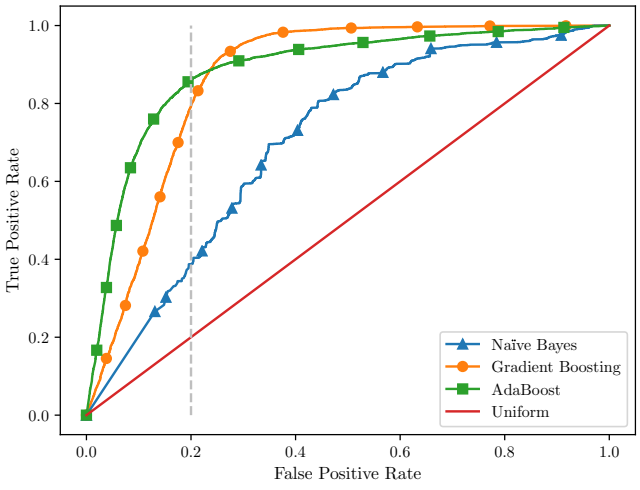

Figure 6: Global ROC curves for recommendations from a full closet.

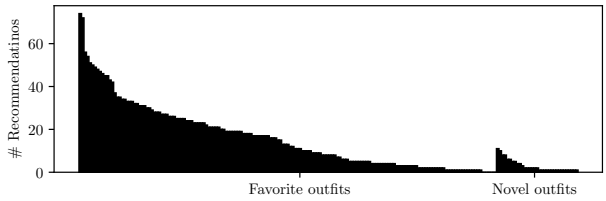

Figure 7: Distributions of outfit recommendations using AdaBoost.

Figure 7 shows the distributions of outfit recommendations in a top-20 list recommended to the users with at least 20 outfit likes. In total, 196 unique outfits were recommended to the users, where 33 of them were novel outfits—never favored by any users in the past. This shows that a wide range of outfits end up in the users' recommended top lists.

Experiment on a half empty closet resulted in no change in terms of overall Recall, and at most, a .005 decrease in AUC, and for this reason, we do not report any results beyond this. Besides the fact that few clothing items will result in fewer outfit recommendations, we conclude that closet size has little effect on the recommendations.

In Table 6, results from evaluation of the conventional approach is given. The table shows Recall@N in a ranked list of M outfits. Because M is much lower than L, we only report up to N = 5 (as

Table 6: Evaluation of the conventional approach.

Model	Recall@1	Recall@5
Cosine	.077	.366
Jaccard	.050	.250

opposed to N = 20 in evaluation of the novel approach). Note that the results are not comparable to the results in Table 5, as they are derived using an approach that is fundamentally different. The best performing model is highlighted with underlined results. As the numbers indicates, the approach generates new outfit recommendations to the users at with a satisfactory accuracy. However, these outfit recommendations are—as argued in Section 5—only outfits that have been composed and favored by other users in the past. Therefore, we conclude that this approach is insufficient when it comes to recommending novel and personalized daily outfits.

8 CONCLUSION AND FUTURE WORK

We have introduced a novel approach for recommending daily fashion outfits from a smart closet. Our novel approach mitigate a wide range of challenges faced by a conventional approach that tries to recommend daily fashion outfits. Evaluation of our novel approach demonstrates the method's effectiveness, and its ability to provide users with accurate and novel outfit recommendations.

The results from the evaluation helped us select which model to deploy in the system. R-score, AUC, and Recall@N are the most useful measures regarding each individual user. Since, AdaBoost achieved the highest R-score and AUC, it was chosen as the main classifier and implemented with the novel approach in the recommender service deployed in the cloud. It should be noted that Gradient Boosting achieved slightly better results in Recall@N, but we regard this difference as insignificant and conclude that AdaBoost is indeed the best fit for our system.

A non-comparable evaluation of the conventional approach was performed to see to what extent it could recommend daily outfits. The accuracy results are acceptable, but due to the approach's many challenges, it cannot be considered as an efficient method for recommending daily outfits.

Although we have demonstrated the system's performance using a real-world dataset, a full scale evaluation using data gathered from physical clothes enabled with RFID tags is planned for future work. The current state of the system should be considered as an early prototype and is premature for such a full scale evaluation. Because of this, these plans are preliminary and we consider other research topics to be more important at the current stage. These topics include content-based outfit recommendation and recommendation of garments to be recycled or donated. With these research topics, we intend to incorporate additional contextual factors such as season, user's occasion, and user's body type.

ACKNOWLEDGMENTS

This work is an extension to a prototype of the proposed system initially developed during an internship at Accenture. The authors would like to thank everyone involved in the internship for their contributions prior this work.

The authors would also like to thank everyone at Accenture who has provided valuable feedback on this research.

REFERENCES

[1] Charu C. Aggarwal. 2016. *Recommender Systems: The Textbook* (1st ed.). Springer Publishing Company, Incorporated.
[2] Arnaud Berlioz, Arik Friedman, Mohamed Ali Kaafar, Roksana Boreli, and Shlomo Berkovsky. 2015. Applying Differential Privacy to Matrix Factorization. In *Proceedings of the 9th ACM Conference on Recommender Systems (RecSys '15)*. ACM, New York, NY, USA, 107–114.
[3] Smriti Bhagat, Udi Weinsberg, Stratis Ioannidis, and Nina Taft. 2014. Recommending with an Agenda: Active Learning of Private Attributes Using Matrix Factorization. In *Proceedings of the 8th ACM Conference on Recommender Systems (RecSys '14)*. ACM, New York, NY, USA, 65–72.
[4] Rich Caruana and Alexandru Niculescu-Mizil. 2006. An Empirical Comparison of Supervised Learning Algorithms. In *Proceedings of the 23rd International Conference on Machine Learning (ICML '06)*. ACM, New York, NY, USA, 161–168.
[5] Thomas G. Dietterich. 2000. Ensemble Methods in Machine Learning. In *Proceedings of the First International Workshop on Multiple Classifier Systems (MCS '00)*. Springer-Verlag, London, UK, UK, 1–15.
[6] Bojana Dumeljic, Martha Larson, and Alessandro Bozzon. 2014. Moody Closet: Exploring Intriguing New Views on Wardrobe Recommendation. In *Proceedings of the First International Workshop on Gamification for Information Retrieval (GamifIR '14)*. ACM, New York, NY, USA, 61–62.
[7] Yoav Freund and Robert E Schapire. 1997. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *J. Comput. Syst. Sci.* 55, 1 (Aug. 1997), 119–139.
[8] Jerome H. Friedman. 2000. Greedy Function Approximation: A Gradient Boosting Machine. *Annals of Statistics* 29 (2000), 1189–1232.
[9] K. N. Goh, Y. Y. Chen, and E. S. Lin. 2011. Developing a smart wardrobe system. In *2011 IEEE Consumer Communications and Networking Conference (CCNC)*. 303–307.
[10] Derek L. Hansen and Jennifer Golbeck. 2009. Mixing It Up: Recommending Collections of Items. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. ACM, New York, NY, USA, 1217–1226.
[11] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. 2004. Evaluating Collaborative Filtering Recommender Systems. *ACM Trans. Inf. Syst.* 22, 1 (Jan. 2004), 5–53.
[12] Kurt Jacobson, Vidhya Murali, Edward Newett, Brian Whitman, and Romain Yon. 2016. Music Personalization at Spotify. In *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys '16)*. ACM, New York, NY, USA, 373–373.
[13] Dietmar Jannach, Lukas Lerche, and Iman Kamehkhosh. 2015. Beyond "Hitting the Hits": Generating Coherent Music Playlist Continuations with the Right Tracks. In *Proceedings of the 9th ACM Conference on Recommender Systems (RecSys '15)*. ACM, New York, NY, USA, 187–194.
[14] Ian Jolliffe. 2002. *Principal component analysis*. Wiley Online Library.
[15] Ingunn Grimstad Klepp and Kirsi Laitala. 2016. *Clothing consumption in Norway*. Technical report 2. Oslo and Akershus University College, Oslo. In Norwegian.
[16] Anders Kolstad, Özlem Özgöbek, Jon Atle Gulla, and Simon Litlehamar. 2017. Connected Closet - A Semantically Enriched Mobile Recommender System for Smart Closets. In *Proceedings of the 13th International Conference on Web Information Systems and Technologies (WEBIST 2017)*. 298–305.
[17] Jure Leskovec, Anand Rajaraman, and Jeffrey David Ullman. 2014. *Mining of Massive Datasets* (2nd ed.). Cambridge University Press, New York, NY, USA.
[18] David D. Lewis. 1998. Naive (Bayes) at Forty: The Independence Assumption in Information Retrieval. In *Proceedings of the 10th European Conference on Machine Learning (ECML '98)*. Springer-Verlag, London, UK, UK, 4–15.
[19] Chantima Limaksornkul, Duangkamol Na Nakorn, Onidta Rakmanee, and Wantanee Viriyasitavat. 2014. Smart Closet: Statistical-based apparel recommendation system. In *Student Project Conference (ICT-ISPC), 2014 Third ICT International*. IEEE, 155–158.
[20] John C Pruit. 2015. Getting Dressed. *Popular Culture as Everyday Life* (2015).
[21] Andrew I. Schein, Alexandrin Popescul, Lyle H. Ungar, and David M. Pennock. 2002. Methods and Metrics for Cold-start Recommendations. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '02)*. ACM, New York, NY, USA, 253–260.
[22] Xiaoyuan Su and Taghi M. Khoshgoftaar. 2009. A Survey of Collaborative Filtering Techniques. *Adv. in Artif. Intell.* 2009, Article 4 (Jan. 2009).
[23] Andreu Vall. 2015. Listener-Inspired Automated Music Playlist Generation. In *Proceedings of the 9th ACM Conference on Recommender Systems (RecSys '15)*. ACM, New York, NY, USA, 387–390.
[24] Lin Yu-Chu, Yuusuke Kawakita, Etsuko Suzuki, and Haruhisa Ichikawa. 2012. Personalized Clothing-Recommendation System Based on a Modified Bayesian Network. In *Proceedings of the 2012 IEEE/IPSJ 12th International Symposium on Applications and the Internet (SAINT '12)*. IEEE Computer Society, Washington, DC, USA, 414–417.